Eye Tracking Analysis of Code Layout, Crowding and Dyslexia - An Open Data Set

Eye Tracking Analysis of Code Layout, Crowding and Dyslexia - An Open Data Set

Ian McChesney
School of Computing
Ulster University
Newtownabbey, United Kingdom
ir.mcchesney@ulster.ac.uk

Raymond Bond
School of Computing
Ulster University
Newtownabbey, United Kingdom
rb.bond@ulster.ac.uk

ABSTRACT

Within computer science there is increasing recognition of the need for research data sets to be openly available to facilitate transparency and reproducibility of studies. In this short paper an open data set is described which contains the eye tracking recordings from an experiment in which programmers with and without dyslexia reviewed and described Java code. The aim of the experiment was to investigate if crowding in code layout affected the gaze behaviour and program comprehension of programmers with dyslexia. The data set provides data from 30 participants (14 dyslexia, 16 control) and their eye gaze behaviour in reviewing three small Java programs in various combinations of crowded and spaced configurations. The key features of the data set are described and observations made on the effect of alternative area of interest configurations. The paper concludes with some observations on enhancing access to data sets through metadata, data provenance and visualizations.

CCS CONCEPTS

• **Software and its engineering** → *Maintaining software*; • **Human-centered computing** → *Empirical studies in accessibility*.

KEYWORDS

data sets, code layout, crowding, dyslexia, eye tracking

ACM Reference Format:
Ian McChesney and Raymond Bond. 2021. Eye Tracking Analysis of Code Layout, Crowding and Dyslexia - An Open Data Set. In *ETRA '21: 2021 Symposium on Eye Tracking Research and Applications (ETRA '21 Short Papers), May 25–27, 2021, Virtual Event, Germany*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3448018.3457420

1 INTRODUCTION

Within the discipline of computer science there is increasing recognition of the value of replicable and reproducible research [Boisvert 2016]. One solution to support these aspirations is for researchers to make their data open, allowing others to examine and replicate their findings. In this short paper a study is described which sought to investigate the effect of code layout and crowding on the reading of program code by programmers with dyslexia. Preliminary results from the study are reported in [McChesney and Bond 2021] where it was shown that there was little interaction effect between program type (crowded, spaced) and programmer type (dyslexia, control) regarding gaze behaviour. In this paper we present the artefacts and data from the experiment as a resource which we hope might lead to further work in this specific area. Our aim is not to present a how-to guide on conducting eye tracking experiments in programming but to contribute to the practice of making experimental protocols and open data sets available to the wider research community.

2 BACKGROUND

There is much debate within the scientific community in general e.g. [Baker 2016] and computer science in particular e.g. [Cockburn et al. 2020] on the need to enhance the conduct and reporting of experimental studies in order to improve the ability to repeat, replicate and reproduce experimental results. Mechanisms for achieving this include the pre-registration of studies and their analysis methods, and increased transparency of experimental artefacts, data and results [TOP 2021].

An important contribution of workshop series such as Eye Movements in Programming e.g. [Begel and Siegmund 2019] is to serve as a forum for presenting and discussing planned studies [McChesney and Bond 2018] as a move towards the notion of study pre-registration - a practice which is becoming more widespread in some disciplines e.g. psychology [Kwasnicka et al. 2020]. Also, the dissemination of existing data sets [Bednarik et al. 2020] strengthens the capability of the eye movements in programming community in its efforts towards greater reproducibility or otherwise of results.

Here we present a data set which, though relatively small in terms of sample size, aims to be comprehensive in its outputs and range of artefacts so as to convey data provenance and facilitate reproducibility, complementing other work already established within the eye movements in programming field [Sharafi et al. 2015], [Sharafi et al. 2020], [Bednarik et al. 2020]. In the next section we outline the rationale for the study and its experimental setup. In Section 4 the data set and related artefacts are described. In such studies, the metrics produced depend on AOI (area of interest) selection. Issues pertaining to AOI configuration are described in Section 5. We conclude in Section 6 with some brief observations on conducting eye tracking studies in programming with a view to open data sharing.

ETRA '21 Short Papers, May 25–27, 2021, Virtual Event, Germany

ACM ISBN 978-1-4503-8345-5/21/05.
https://doi.org/10.1145/3448018.3457420

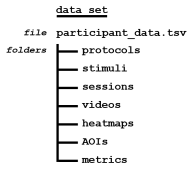

Figure 1: Data set organization

All of the artefacts and raw data described below are available on the figshare data repository [1] under the Creative Commons CC BY license. The data set should be cited by reference to the figshare DOI (Digital Object Identifier) or citation of this paper. The organization of the data set is summarized in Figure 1.

3 EXPERIMENT

3.1 Purpose

The approach described in [McChesney and Bond 2018] outlines the rationale for the present study. Previous work in this area had suggested that programmers with dyslexia, when reading program code, do not exhibit obvious differences in gaze behaviour compared to programmers without dyslexia [McChesney and Bond 2019]. Where differences were apparent, they related to localized parts of the program code. One question to emerge was what role, if any, was played by crowded code layout in the reading performance of the programmer with dyslexia? As noted by [McChesney and Bond 2018], crowding in source code can present in various ways, for example, in Java for loop headers, System.out.format configurations, blocks of sequential statements or failure to comply with style guides.

The present experiment was devised to investigate the interaction effect of crowding and dyslexia. Preliminary results are presented in [McChesney and Bond 2021], focusing on comprehension time, comprehension performance and gaze behaviour as measured by visit count, visit duration, time to first fixation and first visit duration. In summary, the results suggested that there was no significant interaction effect between dyslexia and crowding with respect to comprehension time or performance. With reference to gaze, results also suggested that crowding does not lead to any detrimental reading performance for the programmer with dyslexia, although there are some further questions to explore. For example, (a) is there a tendency for programmers with dyslexia to focus on the on the "edges" of a crowded code space; (b) what is the effect of crowding on the temporal pattern of gaze behaviour for programmers with dyslexia and (c) is there a link between Gestalt principles of program code layout (including crowding) and eye gaze behaviour of programmers with dyslexia. We hope these questions can be addressed by ourselves or others by using or building on this data set.

3.2 Experimental set up

Recruitment was via email to undergraduate computing students at the authors' institution. The invitation email provided both a summary and detailed description of the study. The detailed description constituted the participant information sheet. A £10 Amazon gift voucher was provided as an incentive for participation.

Each session was conducted by a single researcher. At the beginning of a recording session, it was confirmed with the participant that they had read the information sheet as circulated. An opportunity for questions was provided. The researcher guided the participant through the consent form. Then, the preliminary participant questionnaire was completed to collect profile information such as age, programming experience, self-assessment of energy level and, if applicable, information about their dyslexia. On completion of the questionnaire, a brief overview of the eye tracker setup was provided, along with verbal guidance about sitting position and being comfortable. Further details of the experimental configuration are described in Table 1.

The participant was then presented with a sequence of stimuli on screen as follows:

- Welcome screen ("Thank you for agreeing to participate in our study. First we will need to perform some calibration to maximize the accuracy of the eye tracker").
- Tobii X3-120 nine point calibration.

For each program N=1..3:

- Program N introduction ("On the next screen you will see a code snippet. Review this code with a view to understanding its function. When you are satisfied that you understand the code to the best of your ability – say READY").
- Program Na (For review - the participant was encouraged not to think aloud when reviewing the code to minimize researcher provoked data).
- Program Nb (For explanation - when the participant had indicated they now understood the program, the program was shown again and they were invited to describe its purpose and output).
- Self-assessment screen ("How confident are you in your understanding of this code? 1 = Low Confidence 10 = High Confidence").

The full set of stimuli are available in the repository. Program 1 (5 lines of code) is a for loop iterating over the integers 0 to 4 and printing the double of each value. Program 2 (13 lines of code) includes a for loop which iterates over a set of positive and negative integers. On completion of the loop the program prints the sum of the positive numbers and a count of the negative numbers. Program 3 (17 lines of code) prints an array of five integers, then reverses and prints the same array variable. Some subjectivity is involved in determining what constitutes a crowded or spaced version of each program. The aim was to have code layout that would be considered realistic in both formats, hence extremes of crowding or spacing were avoided. For program 1, given its size, horizontal spacing was reduced in the for loop header in both formats of the program to further enhance the crowding effect in the crowded version. For programs 2 and 3, the for loop headers had a space either side of the assignment and relational operators. For each program, the spaced version is characterized by vertical spacing between selected lines of code. Vertical spacing was not introduced between every line due to the realism constraint - some lines remain contiguous to visually retain the logical coherence of the program.

[1] https://doi.org/10.6084/m9.figshare.c.5304473.v2

Table 1: Experimental configuration

Feature	Setting
Typical session duration	approximately 20 minutes
Monitor	Hanns-G HL249DPB 23.6" 1920 x 1080
Eye tracker	Tobii X3-120
Monitor to eye distance	approximately 65 cm
Font size	Program 1 and 2: 27/72 inches (9.53mm: 0.85 degrees) Program 3 18/72 inches (6.35mm: 0.53 degrees)
Software	Tobii Pro Lab v1.145
Actual average calibration (validation) accuracy	0.58 degrees

```
L01 int x = 5;
L02 for (int i=0; i<x; i++) {
L03    int value = i * 2;
L04    System.out.print(value + ",");
L05 }
```

Figure 2: Program 1 - crowded

```
L01 int[] values = {5, -4, 2, -5, 1, 3};
L02 int total = 0, n = 0;

L03 // Setting up for loop
L04 // Loop counter = i
L05 for (int i = 0; i < values.length; i++) {
L06    if (values[i] > 0) {
L07       total = total + values[i];
L08    }
L09    else {
L10       n = n + 1;
L11    }
L12 }
L13 System.out.print("Total: " + total + " n: " + n);
```

Figure 3: Program 2 - crowded

```
L01 int[] numbers = {1, 2, 3, 4, 5};

L02 System.out.println("---------------------");
L03 for (int i = 0; i < numbers.length; i++) {
L04     System.out.print(numbers[i] + " ");
L05 }

L06 int j = numbers.length - 1;
L07 for (int i = 0; i < j; i++) {
L08     int temp = numbers[i];
L09     numbers[i] = numbers[j];
L10     numbers[j] = temp;
L11     j--;
L12 }

L13 System.out.println();
L14 System.out.println("---------------------");
L15 for (int i = 0; i < numbers.length; i++) {
L16     System.out.print(numbers[i] + " ");
L17 }
```

Figure 4: Program 3 - crowded

Each participant was presented with at least one crowded program and one spaced program, organized as six "timelines" in Tobii Pro Lab. These combinations are included in the participant_data file. The crowded versions of the programs are shown in Figures 2-4. Programs 1 and 2 were displayed with a font size of 27/72 inches and program 3 with a font size of 18/72 inches.

4 DATA SET

4.1 Participant Data

The participant data file contains, for each participant: (a) data collected through the preliminary participant questionnaire; (b) the combination of crowded and spaced program types viewed by the participant (their timeline) and, for each program, their comprehension performance (researcher assessed) and self-assessment of performance; (c) viewing time (milliseconds) for each stimulus in the Tobii timeline and (d) Tobii Pro Lab values for actual distance from eye tracker to eye (mm), gaze samples% (the ratio of correctly identified eye-tracking samples to the theoretical maximum), validation accuracy (degrees) and validation precision (degrees).

4.2 Recording data

For each participant, their full recording session data file as generated by the Tobii Pro Lab export feature is provided in tsv (tab separated value) format. The full set of data items available are identified in the repository and defined in the Tobii Pro Lab documentation[2]. The data items in a session file include for each data point: (a) the recording time stamp (milliseconds); (b) eye position coordinates; (c) recording events and stimulus events (e.g program 1 stimulus start, program 1 stimulus end); (d) eye movement type (fixation, saccade) and its duration and (e) AOI hits. In order that further analysis is not limited to the use of Tobii Pro Lab, the files provided are complete data files from which an entire session can be reconstructed.

4.3 Metrics files

For each program type, a file is provided containing the full set of Tobii gaze metrics for each AOI currently designated in the study (further AOIs may be added for future analysis). The AOIs for program 1 are shown in Table 2. AOIs for the other two programs are available in the repository. As such, there is a metrics file for each program 1-3 in both the crowded and spaced configurations.

4.4 Visualizations

For each participant, gaze plot videos (MP4 format) are provided for each program "review" stimulus (*N*a above), giving a total of 90 videos. The average video length is 83 seconds.

Two categories of heat map are also provided, absolute fixation count and absolute fixation duration. For each program, there is a

[2]https://www.tobiipro.com

corresponding heat map showing the gaze metric for (a) all participants; (b) programmers with dyslexia and (c) programmers without dyslexia. Heat maps (fixation count) for program 3 are shown in Figures 5 and 6.

5 AOI CONFIGURATIONS

During the course of an eye movements in programming study, the researcher will make a range of decisions pertaining to stimulus size, AOI definition, metrics selection and other aspects of experimental protocol. An advantage of sharing the study data set is that others can evaluate or explore the effect of alternative choices. One such example is AOI selection and definition.

Even though the Java programs under review are small, they result in a crowded visual space which can lead to problems when defining AOIs. In work involving participants with dyslexia, it is not clear if AOI selection will have an effect on results; for example, there are studies which suggest that, for normal text, dyslexic readers exhibit shorter saccades and more regressions [Bellocchi et al. 2013]. AOI selection therefore merits consideration to ensure optimal coverage and spacing. In preparing the eye tracking data for analysis in the present study, three different AOI configurations were evaluated. These were labelled "strict", "forgiving" and "maximum". The strict configuration set the AOI strictly according to the physical dimensions of the line of code or other program feature of interest. For example, in program 1, this would be an AOI height of 26 pixels and a width of 0 pixels either side of the line of code edges (see Figure 7, yellow shading). The forgiving configuration set the height at 39 pixels (based on an average spacing of 13 pixels between strict AOIs) with 7 pixels of "padding" either side of the line of code edges (pink shading). The maximum configuration set the largest AOIs, ensuring the maximum number of fixations were captured within a designated AOI, yet consistent with the principle of a 1-1.5 degree margin between AOIs [Dalrymple et al. 2018], [Holmqvist et al. 2011] (approximately 30 pixels in this case). For the spaced version of program 1, this policy set an AOI height of 64 pixels with 19 pixel padding either side of the line of code edges (green shading). However, such an arrangement could only be applied to the spaced version of the program since the forgiving configuration already covered white space between lines of code in the crowded program (Figure 8).

The question is which AOI configuration will give the best coverage of the program features of interest while discriminating between gaze pertaining to a selected feature or gaze elsewhere within the program? One way to explore this is to count (a) the number of fixations falling within the set of AOIs and (b) the number of fixations falling outside the set of AOIs but within the bounds of the overall program. Then, looking for any significant difference between the dyslexia and control study groups.

Table 2: Areas of interest

Feature	Area of Interest
Lines of code	L01-L05
Identifiers	value-L03, value-L04
for header	int i=0, i<x, i++

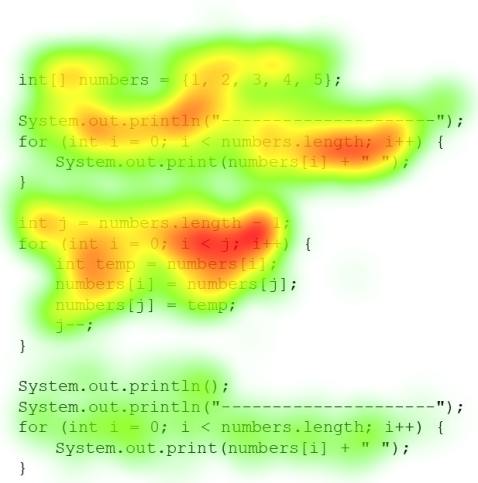

Figure 5: Program 3 crowded heat map (fixation count) - dyslexia participants

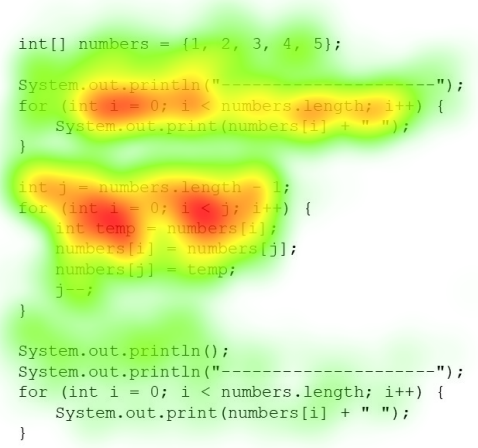

Figure 6: Program 3 crowded heat map (fixation count) - control participants

```
int x = 5;

for (int i=0; i<x; i++) {

   int value = i * 2;

   System.out.print(value + ",");
}
```

Figure 7: Program 1 spaced - LO3 - AOI possibilities (yellow = strict, pink = forgiving, green = maximum)

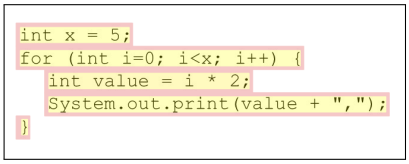

Figure 8: Program 1 crowded - AOI possibilities (yellow = strict, pink = forgiving)

Table 3: Program 1 spaced - contingency tables - whole fixation counts

Policy	AOI	Dyslexia	Control	Total
strict	LOC	*379*	*443*	822
	non-LOC	*321*	*265*	586
	Total	700	708	1408
forgiving	LOC	*557*	*587*	1144
	non-LOC	*151*	*136*	287
	Total	708	723	1431
maximum	LOC	*710*	*726*	1436
	non-LOC	*30*	*28*	58
	Total	740	754	1494

Considering the spaced version of program 1, Table 3 shows contingency tables for whole fixation counts within and not within the set of AOIs (in this case lines of code) . For each configuration, the LOC row shows the number of fixations within the set of AOIs (L01-L05) and the non-LOC row shows the number of fixations not within a line of code AOI yet within the boundaries of the whole program space. (The whole program AOI is not the entire area visible on screen but the rectangular AOI encompassing the particular AOI set. For example, the "strict whole program" area is smaller than the "forgiving whole program" area).

With the strict configuration, 46% of the dyslexia group and 37% of the control group fixations are not counted within the set of AOIs. In other words, of the fixations in the whole program AOI, a greater proportion of the control groups fixations fell within the line of code AOIs. Performing a Chi square test on this difference, there was a significant association between the study groups and whether or not a fixation was on the set of AOIs, χ^2 = 10.29 (df = 1), p = .001. The odds ratio showed that the odds of a fixation being on a LOC were 1.42 times higher if in the control group than if in the dyslexia group.

With the forgiving arrangement, 21% of dyslexia and 19% of control fixations are not counted. For this configuration, there was no significant association between the subject groups and whether or not a fixation was on a LOC or in the program's white space, χ^2 = 1.41 (df = 1), p = .234. The odds ratio in this case was 1.17.

With the maximum configuration, 4% of dyslexia and 4% of control fixations are not counted. As expected following the forgiving results, there was no significant association between the subject groups and whether or not a fixation was on a LOC or in the program's white space, χ^2 = 0.12 (df = 1), p = .733. The odds ratio was 1.10.

It is an interesting observation that with a strict AOI configuration, the gaze of the dyslexia group would appear less concentrated on the central zone of the line of code. However, within the constraints of the font sizes in the study and the accuracy of the eye tracker itself, such an AOI arrangement could indeed ignore fixations which are legitimately to be counted. Given that the maximum configuration leads to overlapping AOIs in the crowded programs, the forgiving configuration of AOIs was adopted for metric generation in the present study.

6 CONCLUSION

There are many benefits to sharing open data sets in computer science research. The eye movements in programming community is an example of this [Bednarik et al. 2020]. The data set presented here seeks to complement and extend such practice. Some observations and recommendations arising from our work are (a) to ensure permission to share data is included during a study's ethical approval stage; (b) in addition to raw data, include derived artefacts such as videos and visualizations to help others understand the data set and lower barriers to access; (c) allow for atypical participants when determining AOIs for data analysis and (d) identify aspects of experimental design where alternatives settings or parameters might be significant and could be evaluated by other researchers.

ACKNOWLEDGMENTS

We are grateful to Dr Victoria Simms for conversations on the study design and ethics approval. And to the participants who consented to their gaze data being used for this and further study.

REFERENCES

2021. *TOP Guidelines.* https://www.cos.io/initiatives/top-guidelines

Monya Baker. 2016. Reproducibility crisis. *Nature* 533, 26 (2016), 353–66.

Roman Bednarik, Teresa Busjahn, Agostino Gibaldi, Alireza Ahadi, Maria Bielikova, Martha Crosby, Kai Essig, Fabian Fagerholm, Ahmad Jbara, Raymond Lister, et al. 2020. EMIP: The eye movements in programming dataset. *Science of Computer Programming* 198 (2020), 102520.

Andrew Begel and Janet Siegmund (Eds.). 2019. *EMIP '19: Proceedings of the 6th International Workshop on Eye Movements in Programming* (Montreal, Quebec, Canada). IEEE Press.

Stéphanie Bellocchi, Mathilde Muneaux, Mireille Bastien-Toniazzo, and Stéphanie Ducrot. 2013. I can read it in your eyes: What eye movements tell us about visuo-attentional processes in developmental dyslexia. *Research in developmental disabilities* 34, 1 (2013), 452–460.

Ronald F Boisvert. 2016. Incentivizing reproducibility. *Commun. ACM* 59, 10 (2016), 5–5.

Andy Cockburn, Pierre Dragicevic, Lonni Besançon, and Carl Gutwin. 2020. Threats of a replication crisis in empirical computer science. *Commun. ACM* 63, 8 (2020), 70–79.

Kirsten A Dalrymple, Marie D Manner, Katherine A Harmelink, Elayne P Teska, and Jed T Elison. 2018. An examination of recording accuracy and precision from eye tracking data from toddlerhood to adulthood. *Frontiers in psychology* 9 (2018), 803.

Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures.* OUP Oxford.

Dominika Kwasnicka, Gill A Ten Hoor, Anne van Dongen, Ewa Gruszczyńska, Martin S Hagger, Kyra Hamilton, Nelli Hankonen, Matti Toivo Juhani Heino, Marie Kotzur, Chris Noone, et al. 2020. Promoting scientific integrity through open science in health psychology: results of the Synergy Expert Meeting of the European health psychology society. *Health psychology review* (2020), 1–17.

Ian McChesney and Raymond Bond. 2018. Gaze behaviour in computer programmers with dyslexia: considerations regarding code style, layout and crowding. In *Proceedings of the Workshop on Eye Movements in Programming.* 1–5.

Ian McChesney and Raymond Bond. 2019. Eye tracking analysis of computer program comprehension in programmers with dyslexia. *Empirical Software Engineering* 24, 3 (2019), 1109–1154.

Ian McChesney and RR Bond. 2021. The Effect Of Crowding On The Reading Of Program Code For Programmers With Dyslexia. In *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*. IEEE. (in press).

Zohreh Sharafi, Timothy Shaffer, Bonita Sharif, and Yann-Gaël Guéhéneuc. 2015. Eye-tracking metrics in software engineering. In *2015 Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 96–103.

Zohreh Sharafi, Bonita Sharif, Yann-Gaël Guéhéneuc, Andrew Begel, Roman Bednarik, and Martha Crosby. 2020. A practical guide on conducting eye tracking studies in software engineering. *Empirical Software Engineering* 25, 5 (2020), 3128–3174.